REMARKS

Reconsideration and withdrawal of all grounds of rejection, and allowance of the pending claims are respectfully requested in light of the amendments and remarks made herein. Claims 1-15 remain pending.

Figure 1a stands objected to for not being designated with a "Prior Art" legend. In response, replacement sheets have been provided with a "Prior Art" legend as indicated by the Examiner. Accordingly, Applicant requests removal of this objection.

Claims 10-12 stand rejected under 35 U.S.C 101 as it is alleged to be directed to non-statutory matter. In response claims 10-12 have been amended to recite "a processor," thus, reciting a physical object. Accordingly, it is respectfully submitted that the amended claims are allowable subject matter under 35 USC §101.

Claims 1, 2, 5-6, 8, 10-11 and 13-15 stand rejected under 35 USC 102(e) as being anticipated by Kush (USP No. 6,874,144). Claims 3, 9 and 12 stand rejected under 35 USC 103(a) as being unpatentable over Kush in view of Werres et al. (USP No. 5,295,264). In response applicants have amended independent claims 1 and 10 to recite the limitations of "wherein the multiprocessing environment includes real-time and non-real-time operating systems." Applicant can find nothing in Kush which teaches that the priority inheritance scheme therein can be used in a multiprocessing environment that includes both real-time and non-real-time operating systems.

Kush is directed to a "method, system and program for implementing priority inheritance in ***an*** operating system." See Abstract. Thus, Kush does not teach a multiprocessing environment as in the present invention. Moreover, such priority inheritance systems are well known and have inherent limitations as further described in the present invention's specification on page 2, line 21 – page 3, line 14, which states"

> One previous solution to this particular problem is to implement a 'priority inheritance', sometimes called 'priority promotion', mechanism where an owning thread temporarily (e.g. the low priority thread) gets a priority equal to the highest priority of the waiting processes/threads. In this way, no intermediate priority process is able to pre-empt the low priority thread and thus delay the waiting time further for the high priority thread. The Priority Inheritance mechanism has two problems as e.g. disclosed in "Missed it! – How Priority Inversion messes up real-time performance and how the Priority Ceiling Protocol puts it right", N. J. Keeling (Real-Time Magazine 99-4, 1999). One problem arises when a high priority thread shares multiple resources with other threads, that causes priority inheritance to take too much time, and another problem may arise if multiple threads share multiple resources with each other, whereby priority inheritance will not prevent a deadlock if threads allocate the resources in the wrong order. A solution, called 'priority ceiling', to these two problems is proposed in Keeling where an owning thread temporarily gets a priority equal to the highest priority of all threads that are allowed to wait for the mutex of the shared resource.
>
> However, a priority inheritance and a priority ceiling mechanism can not always be used for communication via a shared resource between a high priority thread and a low priority thread whereby an intermediate priority thread (having a priority between the high and low priority thread) may prevent the high priority thread from performing it's actions in time thereby possibly rendering real-time application useless, erroneous, etc. Additionally, the option of a priority inheritance or a priority ceiling mechanism may not always be available or supported e.g. due to restrictions imposed by an operating system running the thread(s). As an example, if e.g. two different operating systems are running on a single processor at a given time, where one of the operating systems is a real-time operating system running the high priority thread and the other operating system is a non-real time system running the low priority thread, then the priority of the low priority thread can not be raised high enough to reach the priority of the high priority thread.

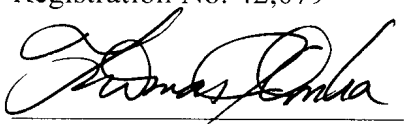
Since Kush does not teach all of the limitations of independent claims 1 and 10, it can not anticipate the present invention. For at least the above cited reasons, Applicant submits that Claims 1 and 10 are patentable over Kush.

With regard to claims 2-9 and 11-15 these claims depend from one of the independent claims discussed above, which have been shown to be allowable in view of the cited reference. Accordingly, each of claims 2-9 and 11-15 are also allowable by virtue of its dependence from an allowable base claim.

For all the foregoing reasons, it is respectfully submitted that all the present claims are patentable in view of the cited references. A Notice of Allowance is respectfully requested.

Respectfully submitted,

Dan Piotrowski
Registration No. 42,079



Date: May 3, 2008

By: Thomas J. Onka
Attorney for Applicant
Registration No. 42,053

Mail all correspondence to:
Dan Piotrowski, Registration No. 42,079
US PHILIPS CORPORATION
P.O. Box 3001
Briarcliff Manor, NY 10510-8001
Phone: (914) 333-9624
Fax: (914) 332-0615